# Polymorphism

Dr. Viet Vo
vvo@swin.edu.au
Department of Computing Technologies
School of Science, Computing and Engineering Technologies

# Learning Outcomes

- The importance of polymorphism in OOP
- Understand how to implement polymorphism
- Demonstrate polymorphism with real-world examples

# Use child objects where the parent is expected

- Polymorphism – a Greek word that means “many-shaped”
- As C# developers, we do not know which specific type of shapes the end-user may want to instantiate in runtime
- No matter which shape selected, it still has a ‘Draw’ behaviour
- Polymorphism can help that with correct objects

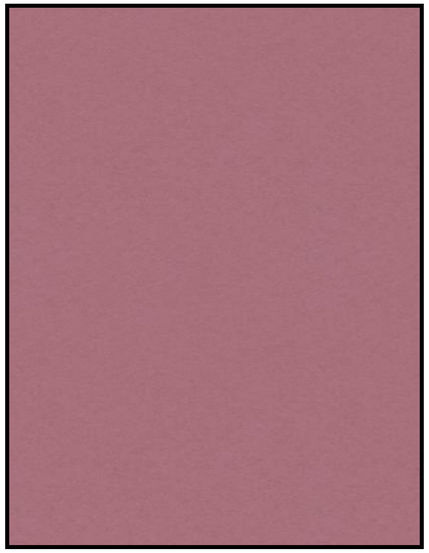
# Refer to an object using any of the classes it **is a** kind of

generalisation

`Object o` Does o refer to an object?

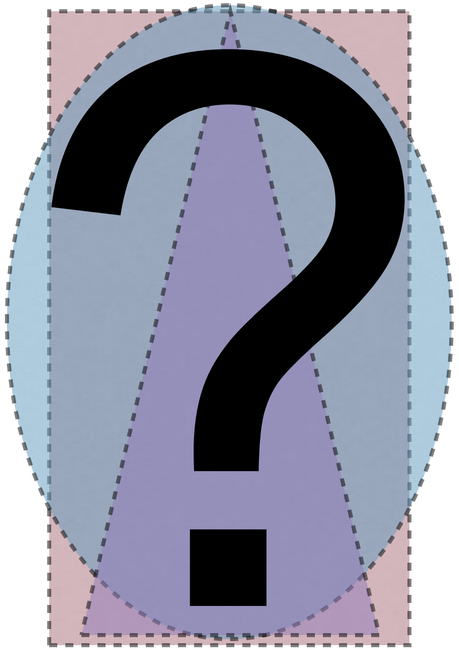
`Shape s` Does s refer to a shape?

`Rectangle r` Does r refer to a rectangle?



# Objects behave based on their **actual** class!

`Shape s` What will be drawn? →



Child class can only have access to :

- public fields and methods of parent class

Otherwise, we must either

- Define new specialised method for the child class
- Override the method defined in the parent class

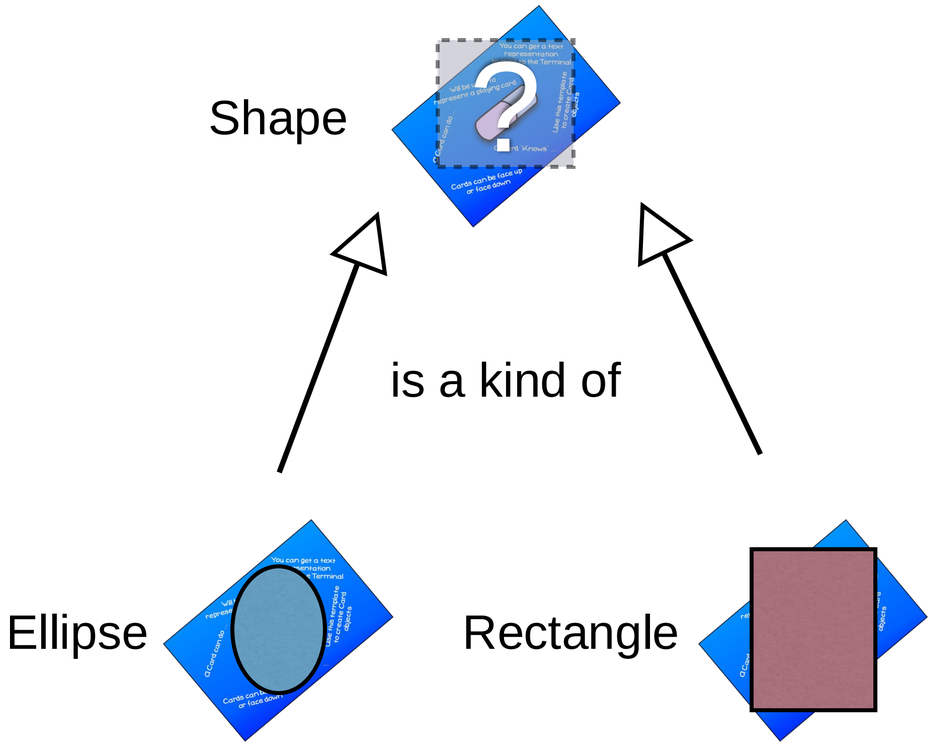
# This is called **polymorphism**

Poly Morph

---

Many Forms

# Parent classes can have *placeholder* methods that **must** be overridden




How does Shape Draw?
It doesn't; Draw is a placeholder = **abstract**
abstract classes **cannot create objects**

Rectangle must override draw

Ellipse must override draw

SWINBURNE UNIVERSITY OF TECHNOLOGY

# Abstract methods of **base classes**

**C++**

virtual void draw () = 0;

**C#**

public abstract void Draw();

**Java**

public abstract void draw();

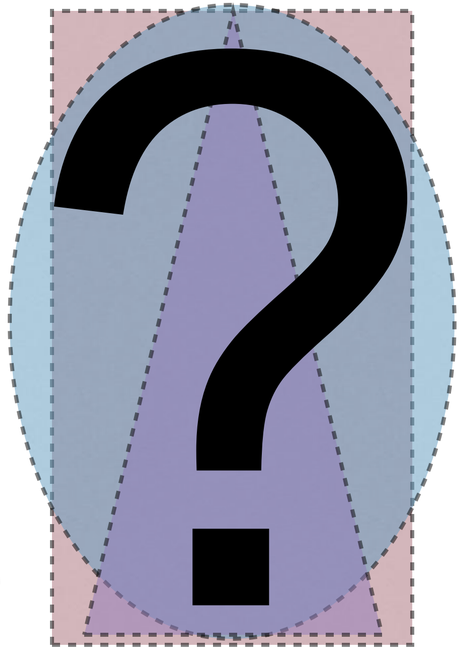
**Objective-C**

- (void) draw;

# How do inheritance and polymorphism help development?

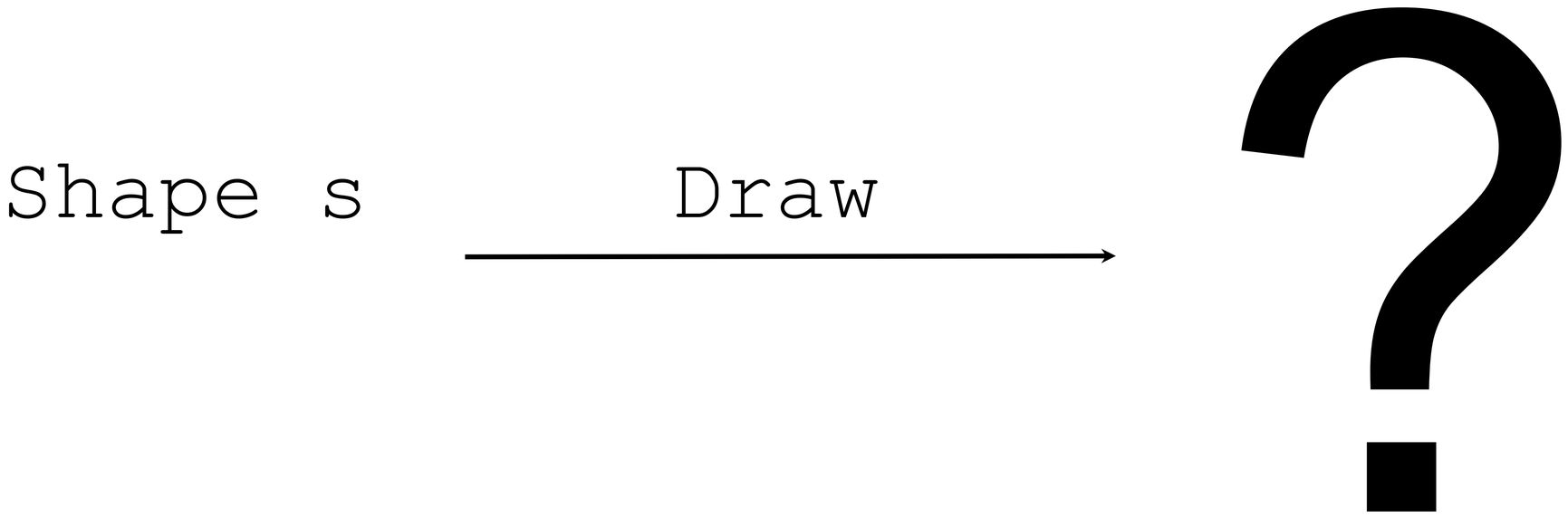
# Flexibility: Refer to a parent class, but get child objects… they work as expected!

`Shape s` `Draw` →



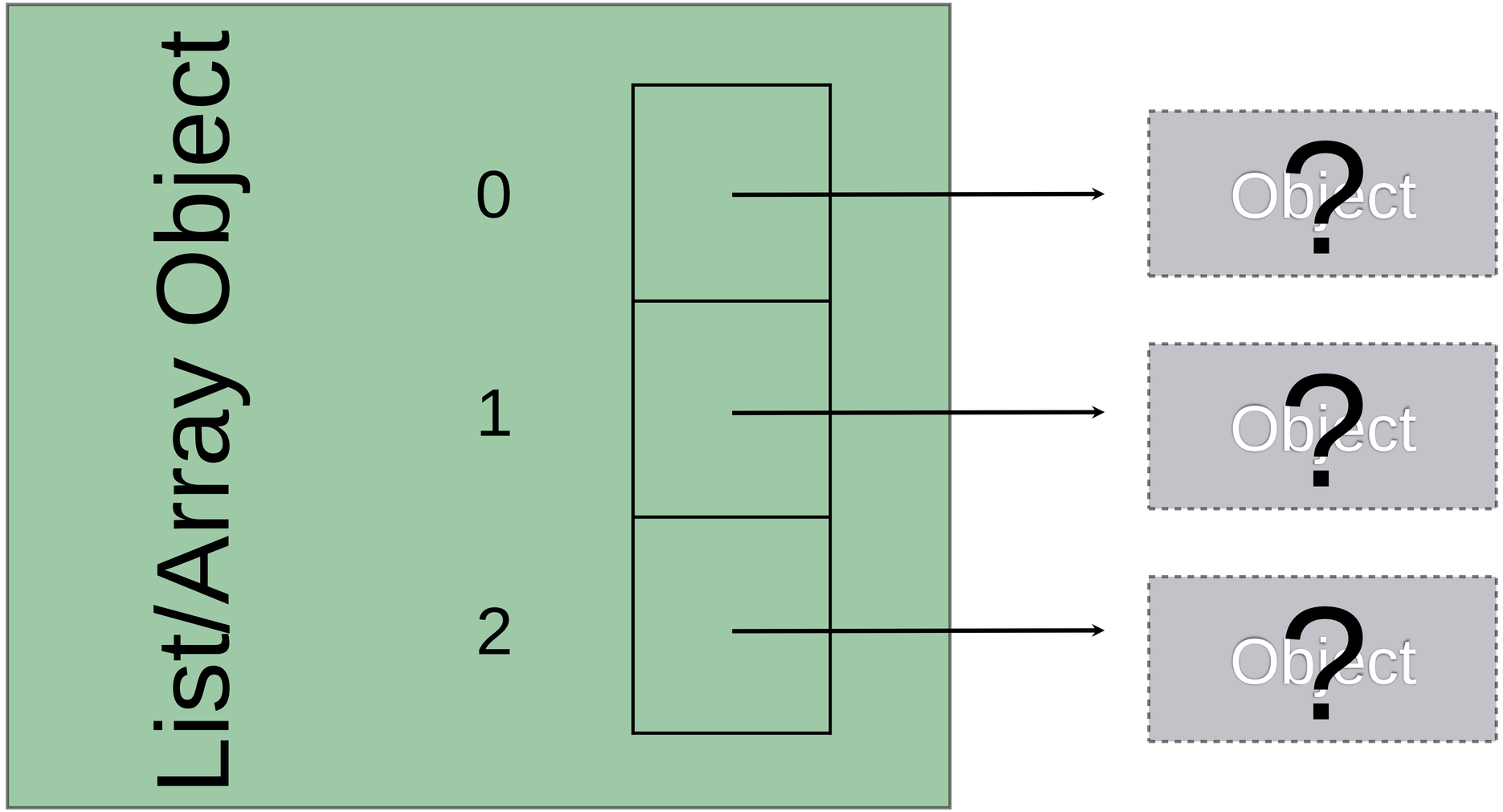
**Shape mywhich = Rectangle/Triangle/Ellipse**
**mywhich.Draw()**

# Extensible: add new children without needing to change uses




Can it only be Rectangle/Triangle/Ellipse?

# Adaptable: Utilities like collection classes can work on Objects



# Take away message

- Polymorphism is the core characteristics of OOP, allowing the program to process different types of objects at different times
- You can redefine methods for child classes
- Polymorphism helps bring flexibility, extensibility, and adaptability to your OO programs
- There are different types of polymorphism and using them in practice is depending on software requirement and your programming skills.